Dirk Praetorius,
Michele Ruggeri

Wintersemester 2015/16
28.10.-30.10.2015

Übungen zur Vorlesung
Einführung in das Programmieren für TM

Serie 3

Aufgabe 3.1. Write a `void`-function `vectorproduct`, which, given two vectors $\mathbf{u} = (a, b, c)^T$ and $\mathbf{v} = (x, y, z)^T$, computes the vector product $\mathbf{w} = \mathbf{u} \times \mathbf{v}$ defined by

$$w_1 = bz - cy$$
$$w_2 = cx - az$$
$$w_3 = ay - bx.$$

Then, write a main program which reads the vectors $\mathbf{u}, \mathbf{v}$ from the keyboard, calls the function and displays the vector product. Save your source code as `vectorproduct.c` into the directory `serie03`.

Aufgabe 3.2. Write a `void`-function `triangle`, which, given three edge-lengths $a, b, c \in \mathbb{R}$ with $a, b, c \geq 0$, determines if the resulting triangle is equilateral, scalene, isosceles, one-dimensional degenerate (the sum of two edges equals the third one) or impossible (the sum of two edges is smaller than the third one). Then, write a main program which reads a, b and c from the keyboard and call the function `triangle`. Save your source code as `triangle.c` into the directory `serie03`.

Aufgabe 3.3. Write a `void`-function `sort3` which gets three real numbers $x, y, z \in \mathbb{R}$ as input. Furthermore, the numbers should be printed out in descending order. Additionally, write a main program that reads in the numbers x, y, z and calls the function. Save your source code as `sort3.c` into the directory `serie03`.

Aufgabe 3.4. Write a `void`-function `roman`, that prints for a given $x \in \mathbb{N}$ with $x \leq 99$ the representation in the roman numeral system. Note, that:

$C \mathrel{\widehat{=}} 100$, $L \mathrel{\widehat{=}} 50$, $X \mathrel{\widehat{=}} 10$, $V \mathrel{\widehat{=}} 5$, $I \mathrel{\widehat{=}} 1$.

Keep in mind the subtraction rule for roman numbers, i.e. write IV instead of $IIII$ for 4. Think about the representation for $x \leq 9$ first, i.e.

I, II, III, IV, V, VI, VII, $VIII$, IX

Then, write down the number of tens in an analogous way. Moreover, write a main-programme that reads x, calls the function `roman`. Save your source code as `roman.c` into the directory `serie03`.

Aufgabe 3.5. The Fibonacci series is recursively defined by $x_0 := 0$, $x_1 := 1$, and $x_{n+1} := x_n + x_{n-1}$. Write the function `fibonacciRec` which returns x_n for given n. Save your source code as `fibonacci.c` into the directory `serie03`.

Aufgabe 3.6. One way (not the best way) the approximate the number π is the so called **Leibniz-formula**:

$$\pi = \sum_{k=0}^{\infty} \frac{4(-1)^k}{2k+1}$$

The n-th partial sum

$$P(n) = \frac{4(-1)^n}{2n+1} + P(n-1)$$

can be interpretated as a recursive function and it holds $\lim_{n\to\infty} P(n) = \pi$. Write a function `double P(int n);` that computes $P(n)$. Moreover, write a main-Programm that reads in $n \in \mathbb{N}$ and computes the n-th partial sum $P(n)$. Hint: You can calculate $(-1)^n$ like in Exercise 2.3. Save your source code as `pirecursive.c` into the directory `serie03`.

Aufgabe 3.7. For $x > 0$, the recursively defined sequence

$$x_1 := \frac{1}{2}(1 + x), \quad x_{n+1} := \frac{1}{2}\Big(x_n + \frac{x}{x_n}\Big) \quad \text{for } n \geq 1$$

converges towards $\sqrt{x}$. Write a recursive function `sqrt_` which computes for given $x > 0$ and $\tau > 0$ the *first* element x_n of the sequence that satisfies

$$\frac{|x_n - x_{n+1}|}{|x_n|} \leq \tau \quad \text{or} \quad |x_n| \leq \tau.$$

Moreover, write a main program which reads in x and τ, computes the approximation x_n of $\sqrt{x}$ and compares it to the exact value, i.e. prints out the absolute error $|x_n - \sqrt{x}|$.
Hint: You can use the function `sqrt` from the math library to compute the exact value $\sqrt{x}$. For the computation of the absolute value $|x|$ of a real number x, you can use the function `fabs` from the math library.

Aufgabe 3.8. Recall the meanings of the terms *Lifetime & Scope*. What is the output of the following code lines?

```
1  #include <stdio.h>
2
3  int max(int,int);
4
5  main() {
6    int x = 1;
7    int y = 2;
8    int z = 3;
9
10   printf("(x,y,z) = (%d,%d,%d)\n",x,y,z);
11
12   {
13     int x = 100;
14     y = 2;
15     z = max(x,y);
16     printf("(x,y,z) = (%d,%d,%d)\n",x,y,z);
17
18     {
19       int z = y;
20       y = 200;
21
22       printf("(x,y,z) = (%d,%d,%d)\n",x,y,z);
23     }
24     printf("(x,y,z) = (%d,%d,%d)\n",x,y,z);
25   }
26   printf("(x,y,z) = (%d,%d,%d)\n",x,y,z);
27 }
28
29 int max(int x, int y) {
30   if(x>=y) {
31     return x;
32   }
33   else {
34     return y;
35   }
36 }
```

Draw a timeline and visualize the lifetime and the scope of the variables `x,y,z`. Moreover, mark all blocks and functions.